>

What's in Your Toolbox?

June 01, 2011

What's in Your Toolbox?

By Fran Lewitter & George Bell

A couple of biologists walk into your office, explaining that they have a pile of data from their big experiment and would like some help with the analysis. They know just what they're looking for, but aren't sure how to get there. How do you choose the right tools for the job? This is probably a common question for many of us, whether we work at the bench or in front of a computer. Unless we're doing something that's novel or extremely specific, we probably have several options for algorithms or software. But which should we select? The choices we're most often asked about include protocols for normalizing microarrays or high-throughput sequencing data sets, identifying differentially expressed genes, or locating peaks from ChIP-chip or ChIP-seq experiments. We have some recommendations, but the answer is often far from straightforward and usually depends on the experimental details. When selecting analysis methods for these, and less common techniques, we've adopted some approaches that may provide pointers for others needing to make such a choice. In all cases, consulting with experts in a specific area of analysis can be a big help, even if they don't agree, because they should be ready to explain why they prefer a certain method — a crucial detail that usually isn't shared in publications. In an ideal case, we can take advantage of a gold standard reference that provides what we can assume to be correct answers. If we start with the corresponding reference data set and process it through any potential methods, we'll get some idea of the accuracy of our results. But even in publications that include comparisons like this, almost invariably some methods are optimal for some groups of experiments — or subsets of experiments — but not for others. And we may not know which group our experiment belongs to.

We almost always concentrate on free, open-source software for our computational biology needs. In the best cases, this model of software development tends to be more agile than commercial software, permitting quicker improvements and bug fixes. On the other hand, with academic software, a single person can make the difference between good and bad support, and software maintenance is assigned very different priorities in different research groups. We're also

glad to use commercial software if it's the standard for our type of analysis or it's known to work better than other options. Few of our typical analysis needs, however, can be paired with standard methods, and even those that are — like Bioconductor for microarray analysis — require many choices among specific algorithms and options.

Make a selection

Of course it's generally helpful to read the publications that accompany new software package releases, even though we have to take these with a grain of salt, given that the authors' novel method almost always comes out on top. We try to choose the best method in an unbiased manner. If it's difficult to decide, we ask whether we should go with a method that's published in the highest-profile publication, by the best-known laboratory, or from the most famous institute. We may find ourselves in the position of any scientist who tries to distill seemingly contradictory studies into the truth. Methods detailed in publications that clearly describe why and how they chose algorithmic details carry an extra advantage compared to those that simply report those details, and especially to those with methods that are too condensed to be reproducible.
[pagebreak]
We may want to put the publications aside and perform our own comparison of techniques to try to decide which works best for an experiment that we've selected as our test data set. This can also be an effective way to decide the right tools for our specific job, although it's surely not a quick way to come to a conclusion. We've performed bake-offs like this and have occasionally found publications taking the same approach. This has the major advantage of being potentially less biased than the usual software comparisons written by software development groups publishing their own code.

So how do we choose the winner — or winners — from our bake-offs? Creating a short-list isn't too hard, as we typically want code that works on Linux, and preferably on Windows and Mac, too. We also want code that accepts our favorite experimental platforms and file formats. Software of the former group is probably optimized for experimental idiosyncrasies, so a system that works great on 454 sequencing may not be so desirable for shorter Illumina reads, and vice versa. Some file format conversions are trivial, but others — like BED versus SAM — describe types of data that can be inherently different. We have also found large variations in the quality and quantity of software documentation, removing some pieces of software from our short list because it wasn't clear to us how they worked. Some methods were clearly designed for the authors' lab; sharing that software was an afterthought. We'll be the first to admit that sharing code with a publication is always better than a brief description in the methods section that's too vague to implement. Nevertheless, poorly documented code isn't much better than no code at all, unless one has the luxury of wading through subroutines to figure out what's going on.

Try them out

Once we come up with a short-list of applications, we can try them all on our test data set. We quickly find that some pieces of software are simply easier to use than others. These tend to be more intuitive, have logical interfaces — whether graphical or command-line — provide helpful warnings and errors, often let us choose the level of detail in output files, and produce attractive figures. We can usually further remove test options that provide only incomplete outputs. If we want to identify differentially expressed genes and are given summary statistics only for those identified, it's not so convincing that the statistics reflect the inherent nature of our biological

system, compared to when we are given similar metrics for every gene or construct. If the output list of differentially expressed genes doesn't include some of our positive controls, for example, it can be difficult to know if we just need to modify our threshold or use a completely different statistical framework. Similarly, for peak-calling in a ChIP-seq experiment, we want to be able to look at genome-wide enrichment scores — like in a wiggle file viewed in a genome browser — to be sure that the peak calls are consistent with the partially processed data.
[pagebreak]
The final major issue we have to face when comparing analysis tools is the choice of parameters and thresholds. We typically start with the defaults, assuming that they're sensible and that the authors have optimized their code with an experiment that's not too different from ours. But if one RNA-seq method has been tested on yeast cells and another on human tumor samples, there's a good chance that variability between these samples gave rise to quite different recommended thresholds for differential expression. The result could easily be that one tool produces an output gene list with many more "interesting" genes than the other. While sensitivity is a good thing, we must also consider specificity. We've found it easier to compare hit lists of similar sizes, so we'll often set tool-specific thresholds, at least for testing. If we have a really high-quality data set — think of a ChIP-seq experiment using a really good antibody that produces highly enriched, narrow peaks — perhaps many methods produce quite similar answers. On the other hand, a traditional task like multiple sequence alignment produces much more variable results as the similarity of the sequences decreases. It's always pleasant when an experiment is robust enough that lots of different ways of analyzing it produce similar results, but at the other end of the continuum we may start to worry if different methods or parameters lead us to different conclusions.

And even once we select a piece of software, choosing the best options is not trivial. But if the options are well described in the documentation, we can choose those based on theory rather than just performance.

Even though we've been doing this for a number of years, we find that maintaining a comprehensive toolbox isn't easy. If we're just tightening a few screws, just about any screwdriver will do. But for complex jobs, we'd like to have a set of tools designed for the tasks at hand that will give us the best possible results.

Fran Lewitter, PhD, is director of bioinformatics and research computing at Whitehead Institute for Biomedical Research. This column was written in collaboration with George Bell, PhD, a senior bioinformatics scientist in Fran's group.

Related Stories

- Is Our Data Any Good?
  March 1, 2011 / Genome Technology
- Juggling Genome Coordinates
  October 1, 2010 / Genome Technology
- Maximizing an Experiment
  June 2, 2010 / Genome Technology
- Bioinformatics and Biologists
  March 1, 2010 / Genome Technology

- How Did You Get That Result?
  August 31, 2009 / Genome Technology